Homework 1

University of Central Florida
Electrical and Computer Engineering

Machine Learning and Pattern Recognition (EEL 5825), Fall 2022

Issued: Wed, Aug 24, 2022 **Due**: Wed, Sep 7, 2022.

1. Consider the sum-of-squares error function given by

$$E(\mathbf{w}) = \frac{1}{2}\sum_{n=1}^{N}\{y(x_n, \mathbf{w}) - t_n\}^2 \tag{1}$$

in which the function $y(x, \mathbf{w})$ is given by the polynomial

$$y(x, \mathbf{w}) = \sum_{j=0}^{M} w_j x^j. \tag{2}$$

Show that the coefficients $\mathbf{w} = \{w_i\}$ that minimize this error function are given by the solution to the following set of linear equations

$$\sum_{j=0}^{M} A_{ij} w_j = T_i \tag{3}$$

where

$$A_{ij} = \sum_{n=1}^{N} (x_n)^{i+j}, \qquad T_i = \sum_{n=1}^{N} (x_n)^i t_n. \tag{4}$$

Here, a suffix i or j denotes the index of a component, whereas $(x)^i$ denotes x raised to the power of i. Find a closed-form expression for the coefficients.

2. Write down the set of coupled linear equations, analogous to (3), satisfied by the coefficients w_i which minimize the regularized sum-of-squares error function given by

$$\widetilde{E}(\mathbf{w}) = \frac{1}{2}\sum_{n=1}^{N}\{y(x_n, \mathbf{w}) - t_n\}^2 + \frac{\lambda}{2}\|\mathbf{w}\|^2, \tag{5}$$

and find a closed-form expression for the coefficients.

3. **(Coding assignment)** In this exercise, you will reproduce the polynomial fitting example we discussed in class and observe the underlying tradeoffs. It is recommended to use an interactive computing environment such as Jupyter Notebook in Python or live scripts in Matlab.

   - **Training data**: generate a training set comprising $N = 10$ observations $\mathbf{x} = (x_1, \ldots, x_N)^T$ and the corresponding values $\mathbf{t} = (t_1, \ldots, t_N)^T$, where the input values x_n for $n = 1, \ldots N$, are spaced uniformly in the range $[0, 1]$ and the target data set $\mathbf{t}$ is obtained by computing the corresponding values of the function $\sin(2\pi x)$ and then adding a small level of random noise with Gaussian distribution $N(0, 0.2)$ with mean 0 and variance 0.2 to each point. In Python, `random.normal` in `NumPy` draws random samples from a normal (Gaussian) distribution. A similar function is `randn` in Matlab.

- **Curve fitting**: Fit a polynomial of order M as in Eq. (2) such that the weights minimize the square error. To this end, you could solve the system of linear equations in Eq. (3) above or use the closed-form expression you derived. Equivalently, you could use `polyfit` in `NumPy` or an equivalent function in Matlab. Plot the training data, the actual model (the sine function), and the learned models. Repeat for $M = 1, 3, 9$. Comment on the complexity of the model and how it leads to overfitting or underfitting. To study the effect of the training data size N, fix a model order (say $M = 9$), then repeat the experiment for different values of N (small and large). Comment on the overfitting as a function of the training set size. Do you see that the same model can behave differently depending on the size of the training data?
- **Model selection**: Compute the root-mean-square (RMS) error E_{RMS} for the training data and for an independent test set. For a set of size N, $E_{\text{RMS}} := \sqrt{2E(\mathbf{w}^*)/N}$, where $\mathbf{w}^*$ are the learned weights and the error $E(\mathbf{w})$ is in Eq. (1). Plot the RMS error as a function of the polynomial degree M. Comment on the tradeoff between the training and testing errors.
- **Overtuning**: In this experiment, we would like to study the effect of the model size on the magnitude of the coefficients. To this end, compute the optimal weights for different values of the model degree M. Comment on the results.
- **Regularization**: The previous experiment motivates the use of regularization where a penalty term is added to the error function to control the values of the weights. In particular, fixing $M = 9$, compute the weights of the polynomial that minimize the regularized error function in Eq. (5). As before, compute the weights and plot the learned model for a non-regularized, moderately regularized and heavily regularized model by setting different values of the regularization parameter λ. Plot E_{RMS} as a function of λ for the training and test sets. Comment on your results.